

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-161509

(43) 公開日 平成8年(1996)6月21日

(51) Int.Cl.<sup>6</sup>

G 0 6 T 11/00  
15/00

識別記号

庁内整理番号

F I

技術表示箇所

9365-5H  
9365-5H

G 0 6 F 15/ 72

3 5 0

4 5 0 A

審査請求 有 請求項の数9 OL (全 15 頁)

(21) 出願番号 特願平6-298618

(22) 出願日 平成6年(1994)12月1日

(71) 出願人 390009531

インターナショナル・ビジネス・マシー  
ズ・コーポレーション

INTERNATIONAL BUSIN  
ESS MASCHINES CORPO  
RATION

アメリカ合衆国10504、ニューヨーク州  
アーモンク (番地なし)

(72) 発明者 川瀬 桂

神奈川県大和市下鶴間1623番地14 日本ア  
イ・ビー・エム株式会社 東京基礎研究所  
内

(74) 代理人 弁理士 合田 潔 (外2名)

最終頁に続く

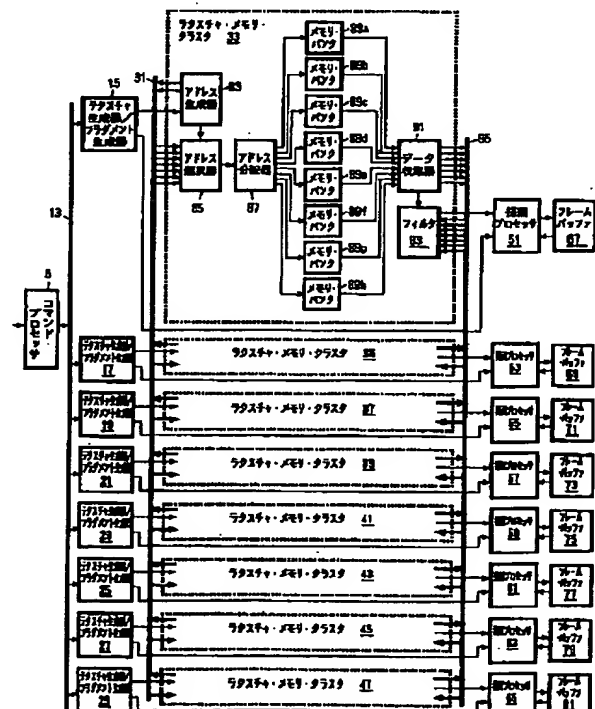
(54) 【発明の名称】 テクスチャ・マッピングを行うコンピュータ・システム

(57) 【要約】

【目的】

【構成】複数のテクセルからなるテクスチャ・イメージ・データをインターリーブして重複がないように各々のメモリに格納する複数のテクスチャ・メモリ・クラスタであって、1のピクセルに対するテクスチャ・イメージの座標から、その座標に対応するテクスチャ値を計算するのに必要なテクセルを求め、そのテクセル値を複数のテクスチャ・メモリ・クラスタから収集する収集手段と、収集したテクセル値から1のピクセルに対するテクスチャ値を計算する手段とをそれぞれ有するものと、複数のメモリ・クラスタを接続するバスと、複数のメモリ・クラスタのうちの1つにそれぞれ接続され、1のピクセルに対するテクスチャ・イメージの座標を計算する複数のテクスチャ生成器とを有する。

【効果】メモリに重複するテクセルを保持することなく、効率的にテクセル値を各メモリ・クラスタ間で交換できる。



BEST AVAILABLE COPY

【特許請求の範囲】

【請求項1】複数のテクセルからなるテクスチャ・イメージ・データをインターリーブして重複がないように各々のメモリに格納する複数のテクスチャ・メモリ・クラスタであって、1のピクセルに対するテクスチャ・イメージの座標から、その座標に対応するテクスチャ値を計算するのに必要なテクセルを求め、そのテクセルの色情報であるテクセル値を前記複数のテクスチャ・メモリ・クラスタから収集する収集手段と、収集した前記テクセル値から前記1のピクセルに対するテクスチャ値を計算する手段とをそれぞれ有する前記複数のテクスチャ・メモリ・クラスタと、

前記複数のテクスチャ・メモリ・クラスタを接続するバスと、

前記複数のテクスチャ・メモリ・クラスタのうちの1つにそれぞれ接続され、前記1のピクセルに対するテクスチャ・イメージの座標を計算する複数のテクスチャ生成器と、

前記複数のテクスチャ・メモリ・クラスタのうちの1つにそれぞれ接続され、前記1のピクセルに対するテクスチャ値を用いて前記1のピクセルに表示すべきデータを生成する複数の描画プロセッサとを有するテクスチャ・マッピングを行うコンピュータ・システム。

【請求項2】前記収集手段が、1のピクセルに対するテクスチャ・イメージの座標から、その座標に対応するテクスチャ値を計算するのに必要なテクセルを求め、前記複数のテクスチャ・メモリ・クラスタに通知する手段と、前記メモリを検索し、通知された前記テクセルのテクセル値を得る検索手段と、前記テクセル値を前記テクセルの通知元のテクスチャ・メモリ・クラスタに送信する手段と、前記テクセル値を受信する手段とを含む請求項1記載のテクスチャ・マッピングを行うコンピュータ・システム。

【請求項3】前記メモリは複数のバンクに別れており、各バンクは、前記テクスチャ・メモリ・クラスタが格納するテクスチャ・イメージ・データをインターリーブして重複を生じないように格納することを特徴とする請求項1又は2記載のテクスチャ・マッピングを行うコンピュータ・システム。

【請求項4】前記複数のバンクは、入力FIFOバッファと、出力FIFOバッファとを含む請求項3記載のテクスチャ・マッピングを行うコンピュータ・システム。

【請求項5】前記検索手段は、前記通知されたテクセルのアドレスを計算する手段と、計算された前記アドレスから該当する前記メモリのバンクに前記アドレスを分配する分配手段と、前記メモリのバンクからテクセル値を得る手段とを含む請求項3又は4記載のテクスチャ・マッピングを行うコ

ンピュータ・システム。

【請求項6】前記必要なテクセルの通知元のテクスチャ・メモリ・クラスタのIDを記憶するFIFOバッファを前記テクスチャ・メモリ・クラスタの各々がさらに有することを特徴とする請求項2乃至5のいずれか記載のテクスチャ・マッピングを行うコンピュータ・システム。

【請求項7】前記計算されたアドレスを分配した前記メモリのバンクのIDを記憶するFIFOバッファを前記テクスチャ・メモリ・クラスタの各々がさらに有することを特徴とする請求項5記載のテクスチャ・マッピングを行うコンピュータ・システム。

【請求項8】前記収集手段が、前記1のピクセルに対するテクスチャ・イメージの座標を前記複数のテクスチャ・メモリ・クラスタに通知する手段と、

前記座標から、その座標に対応するテクスチャ値を計算するのに必要なテクセルであって、自己の前記メモリが保持するテクセルを計算する手段と、

前記テクセルのテクセル値を前記メモリから取り出す手段と、

前記テクセル値を前記テクセルの通知元のテクスチャ・メモリ・クラスタに送信する手段と、

前記テクセル値を受信する手段とを含む請求項1記載のテクスチャ・マッピングを行うコンピュータ・システム。

【請求項9】複数のテクセルからなるテクスチャ・イメージと、そのテクスチャ・イメージを1以上の段階で縮小したデータであり且つ1又は複数のテクセルからなる縮小テクスチャ・イメージとを含むテクスチャ・イメージ・データをインターリーブして重複を生じないよう各々のメモリに格納する複数のテクスチャ・メモリ・クラスタであって、1のピクセルに対する前記テクスチャ・イメージの座標及びピクセルと前記テクスチャ・イメージのテクセルの縮小率から前記1のピクセルに対するテクスチャ値を計算するのに必要なテクセルを求め、そのテクセルの色情報であるテクセルの値を前記複数のテクスチャ・メモリ・クラスタが出力するように要求する手段と、要求された前記テクセルの値を検索し、要求した前記テクスチャ・メモリ・クラスタに出力する手段と、前記テクセルの値を収集する手段と、収集された前記テクセルの値から前記テクスチャ値を計算する手段とをそれぞれ有する前記複数のテクスチャ・メモリ・クラスタと、前記複数のテクスチャ・メモリ・クラスタを接続するバスと、前記複数のテクスチャ・メモリ・クラスタのうちの1つにそれぞれ接続され、前記1のピクセルに対するテクスチャ・イメージの座標及び前記縮小率を計算する複数のテクスチャ生成器と、

10

20

30

40

50

前記複数のテクスチャ・メモリ・クラスタのうちの1つにそれぞれ接続され、前記1のピクセルに対するテクスチャ値を用いて前記1のピクセルに表示すべきデータを生成する複数の描画プロセッサとを有するテクスチャ・マッピングを行うコンピュータ・システム。

#### 【発明の詳細な説明】

##### 【0001】

【産業上の利用分野】本発明は、コンピュータ・グラフィクス装置にかかり、特に、コンピュータを用いて所定の領域へ模様等を張り付けるテクスチャ・マッピングを用いたコンピュータ・グラフィクス装置に関する。

##### 【0002】

【従来の技術】コンピュータ・グラフィクスでは、様々な模様、すなわちイメージ(Texture, テクスチャ)を張り付けるテクスチャ・マッピング(Texture Mapping)によって画像を形成することがある。テクスチャ・マッピング(Texture Mapping)とは、滑らかな平面や曲面上に、木目やスイカの表面の縞模様等の厚みのないパターンやイメージ(Texture)を張ること(Mapping, 以下、マッピングという。)である。例えば、フライト・シミュレータ(米国マイクロソフト社の商標)は、予め撮影しておいた景色の写真イメージを背景部分にテクスチャ・マッピングすることで、仮想現実的な映像を高速で提供している。

【0003】しかしながら、このテクスチャ・マッピングは大量のデータ・アクセス及び演算量を必要とするので、実時間処理するためには、広いバンド幅を有するテクスチャ・メモリ及び専用のハードウェアを必要としていた。このテクスチャ・メモリは、張り付けるべきパターンや背景等のイメージを表す2次元の配列データ(以下、テクスチャ・イメージ・データという。)を記憶するためのメモリである。このテクスチャ・メモリは、高速なアクセスを必要とすると共に大容量を必要とするために、コスト高であった。

【0004】近年のコンピュータの大容量化、小型化及び高速化の加速に伴って、コンピュータ・グラフィクスを利用するコンピュータ・システムでは、高速かつ高効率にテクスチャ・マッピングが可能なハードウェアを低価格で提供することが要求されると共に、課題となっている。以下、テクスチャ・マッピングについて詳述する。

【0005】[テクスチャ・マッピングにおける座標系]コンピュータ・グラフィクスでは、次の3種類の座標系を用いて物体を表現することが一般的である。

【0006】第1の座標系は、モデル座標系(Model coordinate system)である。このモデル座標系は、個々の物体に固有な座標系、すなわち、物体を記述するのに適宜選択される座標系である。図5(A)には、一例として、X軸、Y軸、Z軸の各々が直交する直交座標系における、物体としての立方体OBを示した。

【0007】第2の座標系は、視点座標系(Eye coordinate system)である。この視点座標系は、物体を実質的に目視した時の目の位置に合致された座標系、すなわち、図5(C)に示すように、視線と1つの座標軸とが合致する座標系である。

【0008】第3の座標系は、スクリーン座標系(Screen coordinate system)である。このスクリーン座標系は、図5(D)に示すように、物体を含む3次元のシーンが投影された2次元のスクリーンSC(図5(C)参照)における座標系である。図5(C)におけるスクリーン座標系は、直交する横軸のX<sub>SC</sub>軸、及び縦軸のY<sub>SC</sub>軸によって構成されている。

【0009】テクスチャ・マッピングにおいては、上記の3種類の座標系の他に、図5(B)に示すように、テクスチャ・イメージ・データを記述するための座標系(以下、テクスチャ座標系:Texture coordinate system, という。)が必要である。図5(B)におけるテクスチャ座標系は、直交する横軸のU軸、及び縦軸のV軸によって構成されている。テクスチャ・マッピングの実質的な動作は、スクリーン座標系で本来の物体の色を描画すべき点に、テクスチャ座標で指定されるテクスチャ・イメージ・データの色を描画することである。

【0010】上記のモデル座標系は、複数の基本的物体を組み合わせた複雑な物体を扱うときに、便宜上、各基本的物体を簡単に記述するための座標系である。従って、複雑な物体に対してテクスチャ・マッピングをしてスクリーン座標系で記述するためには、これら複数の基本的物体を纏めた複雑な物体を視点座標系に直す。ここでは、立方体OBに対して、上面にシェーディング(Shading)をすると共に、側面にテクスチャ・マッピングする場合を説明する(図5(C)の立方体OB1参照)。この後に、記述された複数の基本的物体をスクリーンSCに投影することを想定する。この時点では、複雑な物体(立方体OB1)はスクリーン座標系で記述される。なお、以下の説明を簡単にするため、取り扱う物体は、既にスクリーン座標系に変換されているものとする。

#### 【0011】[テクスチャ・マッピングの詳細]図6

(A)に示すように、テクスチャ・イメージ・データは、平面上に単位正方形(1辺の長さが1の正方形)が縦横に配置され、所定の大きさの正方形(図6(A)では、8×8の単位正方形で形成される正方形領域)を形成している。この格子点、すなわち、i番目の列( $0 \leq i \leq 7$ )でj番目の行( $0 \leq j \leq 7$ )に対応する格子点(図中の×マーク)を中心とする各々の単位正方形の中心には、張り付けるべきイメージの色情報が格納されている。

【0012】以下、これらの各単位正方形をテクセル、単位正方形の中心をテクセル中心、テクセル中心に格納されている色情報をテクセル値、及びテクセルが配置さ

れた所定の大きさの正方形領域をテクスチャという。また、テクスチャの一辺に含まれるテクセルの個数(図6(A)では、8個)をテクスチャのサイズという。以下の説明を簡略にするため、j番目の行でかつi番目の列のテクセル(i, j)のテクセル値をT[i, j]と略記する。

【0013】次に、テクスチャを曲面に張り付ける場合、通常の曲面はポリゴン分割されており、視点座標系ではテクスチャ・マッピングとはポリゴンとテクスチャの間の線形変換になる。これによって、ポリゴン上の各点とテクスチャ内の各点は厳密に1対1で対応し、さらに、この対応はテクスチャをポリゴンに張り付ける時点で完全に決定できる。最終的には、画面上(スクリーン上)に投影されたポリゴン内の各ピクセルの色を決定しなければならない。従って、スクリーン座標系は視点座標系の各ピクセルとテクスチャ内の各テクセルの間で対応しなければならないが、スクリーン座標系は視点座標系をスクリーンSC上に投影したものであるため、ピクセル(x\_s, y\_s)に対応するテクスチャ内の点P(u, v)の座標は定まっている。この対応を用いてピクセル(x\_s, y\_s)の色を決定する。一般に用いられている色決定のアルゴリズムは、オリジナルのテクスチャ・イメージ・データと以下に説明するミップマップ(Mipmap)と呼ばれるオリジナル・データをさらに加工したデータを用いる方法、及びミップマップを用いずにオリジナルのテクスチャ・イメージ・データ等を用いる方法に大別される。

【0014】ミップマップを用いない方法には2つの方 \*

$$(1-a) \times (1-b) \times T[i_0, j_0] + (1-a) \times b \times T[i_0, j_0+1] \\ + a \times (1-b) \times T[i_0+1, j_0] + a \times b \times T[i_0+1, j_0+1]$$

引数 i\_0, j\_0 は u, v の整数部分として求めることができる。他の方法は式から理解されるように線形補間を2回続けた行った場合と等価であるのでバイリニアともいう。

【0017】次に、ミップマップを用いる方法を説明する。上記の方法では、投影されたポリゴン上の各点におけるテクセルの縮小拡大率がピクセルの位置によって大きく変化するので、スクリーンとのなす角度が大きなポリゴンが投影された時に不都合を生ずる。すなわち、スクリーン上極端に縮小されるテクセルに対しては上記のように4テクセルの重みつき平均では足りず、多くのテクセルが必要である。この必要とするテクセルは、ピクセルとテクセル間の縮小拡大率(Level Of Detail, ピクセルとテクセルの対応関係から演算できる。線型補間計算を行う時に必ず生ずる。以下、LODという。)から求めることができる。このようにピクセル毎に平均のとり方を変化させる方法(コンポリューションと呼ばれる方法)は効果的だが、効率的でなく、実装が困難であるので以下に説明するミップマップを用いて代用するのが一般的である。

\*法があり、その1つは点Pを含むテクセルのテクセル値をピクセルの色とする方法である。すなわち、点Pに最も近いテクスチャ内のテクセル(i\_0, j\_0)のテクセル値T[i\_0, j\_0]をピクセルの色とする。引数 i\_0, j\_0 は各々 u + 0.5, v + 0.5 の整数部として求めることができる。これは3次元グラフィックス・インタフェースのAPI(Application Program Interface)の1種であるOpenGL(Mar Segal, Akeley, The OpenGL Graphics System: A Specification Version 1.0)では、テクセル・ニアレスト(Texel Nearest)と呼ばれる方法である。

【0015】しかしながら、テクスチャ・マッピングされるポリゴンが近くにあり、スクリーン座標系上では非常に拡大される場合、1つのテクセル値が多数のピクセルに対応することになるので、タイルをはったようになり、ギザギザを生ずる。

【0016】他の方法は、上記を改良した第2のテクセル・リニア・インタポーレイテッド(Texel Linear Interpolated、略して、Texel Linearともいう。以下、テクセルリニアという)である。この場合、図6(A)に示すように、点Pを囲む4つのテクセルの値を線形補間するというものである。すなわち、点P(u, v)を囲むテクセル中心を(i\_0, j\_0), (i\_0+1, j\_0), (i\_0, j\_0+1), (i\_0+1, j\_0+1)

(図中の太線の×マーク)としてa=uの少数部分、b=vの少数部分とおいたときに、以下の式で得られる値をピクセルの色とするものである。

※【0018】ミップマップは、先ずオリジナルのテクスチャ中のテクセルを、図6(A)の太線で囲まれる縦横2個ずつのテクセルをブロックに纏める。次に、ブロック内に含まれる4つのテクセルのテクセル値を平均することによってブロックのテクセル値を求める。次に、図6(B)に示すように、求めたテクセル値に対応させるための新しい1つのテクセルを生成すると共に、元のテクスチャにおけるブロックの順序に配列し、全てのブロックについて行うことによって新しいテクスチャを生成する。このブロックで纏めることにより生成された新しいテクスチャと、生成前のオリジナル・テクスチャとを区別するために、オリジナル・テクスチャ(図6

(A))をレベル0のミップマップ、そしてここではサイズが半分になった新しいテクスチャ(図6(B))をレベル1のミップマップという。次に、レベル1のミップマップを同様に処理してレベル2のミップマップを生成する。この生成されたテクスチャが、サイズ1のテクスチャになるまで上記と同様の処理を繰り返し、高レベルのミップマップを生成する。なお、このときのテクセル値がミップマップのレベルに依存することは勿論であ

る。これを表すため上記の記号を拡張してレベルkのミップマップ中のテクセル (i, j) のテクセル値を  $T[i, j : k]$  と表記する。

【0019】このミップマップを使ってテクスチャ・マッピングするのに先だって、上記のLODから用いるべきミップマップのレベルを決定しなければならない。OpenGLでは細かい規定がなされているが、ここでは必要な部分のみに単純化して説明する。上記のテクセル・ニアレスト、テクセル・リニアと同様にミップマップ・ニアレスト、ミップマップ・リニアの概念があり、合計4種類のテクセル・ミップマップ法がある。なお、各レベルのミップマップ中のピクセルに対応する点の座標P

(u, v) はレベルkに依存したP (u<sub>k</sub>, v<sub>k</sub>) となるので、注意しなければならない。

【0020】次に、テクセル・ミップマップ法の4種類1から4の各々を説明する。

1. ニアレスト・ミップマップ・ニアレスト (Nearest-Mipmap-Nearest)

$k = (\text{LOD} + 0.5 \text{ の整数部分})$  としてレベルkのミップマップ中にピクセルに対応する点P<sub>k</sub> (u, v) を求め、点P<sub>k</sub>を含むテクセル値  $T[i_0, j_0 : k]$  をピクセルの色とする。

2. リニア・ミップマップ・ニアレスト (Linear-Mipmap-Nearest)

$k = (\text{LOD} + 0.5 \text{ の整数部分})$  としてレベルkのミップマップ中にピクセルに対応する点P<sub>k</sub> (u, v) を求め、点P<sub>k</sub>を囲む4つのテクセルのテクセル値を線形補間する。これは「レベルk」ということ以外は上記で説明したテクセルリニアと同様である。

3. ニアレスト・ミップマップ・リニア (Nearest-Mipmap-Liner)

$k = (\text{LODの整数部分})$  としてレベルkとレベルk+1の各々のミップマップ中のピクセルに対応する点P<sub>k</sub>, P<sub>k+1</sub>を各々含むテクセルのテクセル値を  $d = (\text{LODの少数部分})$  を内分比として線形補間する。

4. リニア・ミップマップ・リニア (Linear-Mipmap-Liner)

$k = (\text{LODの整数部分})$  とする。レベルkのミップマップ中のピクセルに対応する点P<sub>k</sub>を含む4つのテクセルのテクセル値を線形補間した(テクセルリニア)結果と、レベルk+1のミップマップ中のピクセルに対応する点P<sub>k+1</sub>を含む4つのテクセルのテクセル値を線形補間した(テクセルリニア)結果を、 $d = (\text{LODの少数部分})$  を内分比として線形補間(ミップマップリニア)する。

【0021】以上説明したようなミップマップ法を用いて、高速にテクスチャ・マッピングを行うためのコンピュータ・グラフィクス装置には、次の点が要求される。

- ・テクスチャ・メモリの容量が大きいこと
- ・テクスチャ・メモリとして用いるメモリのアクセス速

度が高速なこと

- ・テクスチャ・メモリを効率的に利用すること

【0022】容量の点を考えると、テクスチャ・メモリはテクスチャのテクスチャ・イメージ・データを格納する関係上、数Mバイトの容量が必要である。このため、テクスチャ・メモリには一般的に低コストのDRAMが用いられる。

【0023】そして速度の点を考えると、このDRAMのサイクルタイムは80ns程度なので、12.5Mテクセル/秒程度で動作させることができる。但し、上記で述べたようにリニア・ミップマップ・リニア方式では同時に8つのテクセル値を参照しなければならない。8つのテクセル値を同時に参照するには、ミップマップのレベル毎にインターリーブをかけ、さらに各々のレベルをi, jごとにインターリーブして8つのメモリに格納してリニア・ミップマップ・リニアに必要なテクセル値を同時に参照できるような構成にしなければならない。そうしないと、100Mテクセル/秒程度のアクセス速度を出すことができない。

【0024】また、線形補間の計算をする回路(補間回路、Interpolator)が8つのメモリチップに直結している必要がある。各メモリチップに対するアドレスバスと各メモリチップから補間回路へのデータバスのビット幅は、アドレスが20ビット、テクスチャ・イメージ・データが16ビットとしても、 $(20+16) \times 8 = 288$ ビットとなり、配線が複雑化すると共に、チップのピンネック(1チップから出すことのできる信号線の制限)という問題も発生する。

【0025】[従来例]上記の要求を満たす一例として、コンピュータ・グラフィクス装置に対応するテクスチャ・マッピング処理方法がある(特開昭62-140181号公報参照)。このテクスチャ・マッピング処理方法では、プロセッサ素子内部にテクスチャ・イメージ・データを分割して有し、プロセッサ間通信路を介して他のプロセッサ素子内部のテクスチャ・イメージ・データを受けとっている。

【0026】しかしながら、このコンピュータ・グラフィクス装置では、次のような問題がある。

・基本的にソフトウェアによって処理するための機構であり、動作が複雑でハードウェア化が困難であり、高速動作が期待できない。

・描画を処理する部分とテクスチャを処理する部分が1対1に対応しているので、各々の比率を変化できない。このため、描画プリミティブの種類や大きさ、それに貼付けるテクスチャイメージの大きさに対する、パフォーマンス要求に対して、テクスチャを処理する部分の量を変化させることができない。

・他のプロセッサ素子に対してプロセッサ間通信路を介して、表示すべき画素の情報を送り、その送信した情報に基づく演算結果を受けとるというプロトコルなので、

プロトコルのオーバーヘッドが大きい。また、ミップマップを用いた方法を直接適用しようとする、テクセルを全く重複しないように保持することができないので、各プロセッサ内のメモリの容量が大きくなければならない。

【0027】また、他例であるコンピュータ・グラフィクス装置として、リアリティ・エンジン・グラフィクス (Reality Engine Graphics、COMPUTER GRAPHICS Proceedings, Annual Conference Series, 1993pp-116 参照) がある。

【0028】図7に示すように、このコンピュータ・グラフィクス装置90は、コマンド・プロセッサ12を備えている。コマンド・プロセッサ12は、コマンド・バス16を介してフラグメント生成器 (Fragment Generator) 72、74、76、78に接続されている。

【0029】フラグメント生成器72の出力側の一方はテクスチャ・メモリとしてのテクスチャ・メモリ・クラスタ (TEXTURE MEMORY CLUSTER) 22を介して描画プロセッサ (Drawing Processor) 42に接続されており、出力側の他方は描画プロセッサ42に直接接続されている。描画プロセッサ42はフレーム・バッファ52に接続されている。このフレーム・バッファ52の出力側は、ディスプレイ60に接続されている。テクスチャ・メモリ・クラスタ22は、アドレス生成器62、メモリ (メモリ・バンク) 64A、64B、64C、64D、64E、64F、64G、64H及びフィルタ66から構成されており、アドレス生成器62は各メモリ64A～64Hに接続され、各メモリ64A～64Hはフィルタ66に接続されている。

【0030】また、フラグメント生成器74の出力側の一方はテクスチャ・メモリ・クラスタ24を介してフレーム・バッファ54に接続された描画プロセッサ44に接続されており、他方は描画プロセッサ44に直接接続されている。同様に、フラグメント生成器76の一方はテクスチャ・メモリ・クラスタ26を介してフレーム・バッファ56に接続された描画プロセッサ46に接続されており、他方は描画プロセッサ46に直接接続されている。フラグメント生成器78の出力側の一方はテクスチャ・メモリ・クラスタ28を介してフレーム・バッファ58に接続された描画プロセッサ48に接続されており、他方は描画プロセッサ48に直接接続されている。また、上記のフレーム・バッファ54、56、58の出力側は、ディスプレイ60に接続されている。

【0031】なお、テクスチャ・メモリ・クラスタ24、26、28は、テクスチャ・メモリ・クラスタ22と同様の構成のため、詳細な説明を省略する。

【0032】また、このコンピュータ・グラフィクス装置90では、スクリーンについて縦方向を4列に分割して、分割された各々に属するピクセルをテクスチャ・メモリ・クラスタ22、24、26、28が担当し処理し

ている。

【0033】次に、コンピュータ・グラフィクス装置90の動作を説明する。まず、コマンド・プロセッサ12は、その下流にあるフラグメント生成器72～78、テクスチャ・メモリ・クラスタ22～28、及び描画プロセッサ42～48の各プロセッサに対する命令を解釈すると共に、その命令等をコマンドバス16に出力する。この命令には、各々のプロセッサ42～48中の図示しないレジスタにデータを格納するコントロール・コマンドや描画コマンドがある。スクリーン上における描画は、三角形単位 (ポリゴン単位) で行われる。コマンド・プロセッサ12は、この描画コマンドを受け取ると、それに続く三角形の頂点単位のデータ (三角形の頂点座標等) を受け取る。このデータのフォーマットを以下の図8に示す。

但し、 $x\_s$ ,  $y\_s$  : 三角形の頂点のスクリーン座標系での座標

$R$ ,  $G$ ,  $B$ ,  $A$  : スクリーンSC上の座標 ( $x\_s$ ,  $y\_s$ ) における色情報

20  $u$ ,  $v$  : 座標 ( $x\_s$ ,  $y\_s$ ) にテクスチャ・マッピングすべきテクスチャ・イメージ・データのテクスチャ座標系における座標

【0034】これらのデータは、フラグメント生成器72、74、76、78の各々に、コマンドバス16を介して同時にブロードキャストされる。各フラグメント生成器は、得られたデータを補間して三角形内部のピクセル単位のデータを生成する。このとき生成されたピクセルのデータが、フラグメント生成器自身が担当するピクセルのデータであれば、ピクセル単位に補間されたデータ (以下の図9に示す) を、対応する描画プロセッサ42～48に出力する。

【0035】これと共に、フラグメント生成器は、以下の図10に示すデータ、すなわち、三角形の頂点のスクリーン座標系での座標、及び座標 ( $x\_s$ ,  $y\_s$ ) にテクスチャ・マッピングすべきテクスチャ・イメージ・データのテクスチャ座標系における座標 ( $u$ ,  $v$ ) を、対応するテクスチャ・メモリ・クラスタに出力する。

【0036】このとき、フラグメント生成器自身が担当するピクセルでない場合は、描画プロセッサにおいてデータは破棄されるが、ピクセルは必ず何れかのフラグメント生成器が担当しているので、全く処理されないピクセルが生じることはない。テクスチャ・メモリ・クラスタのアドレス生成器は、この座標 ( $u$ ,  $v$ ) に基づいて、上述した例えばリニア・ミップマップ・リニアの計算に必要なテクセルを求め、そのテクセルのメモリにおける物理アドレスを計算し、出力する。この物理アドレスをもとにメモリはテクセル値をフィルタに出力し、フィルタは出力されたテクセル値をもとに例えばリニア・ミップマップ・リニアに必要な計算を行って、ピクセルに張り付けるべき、テクスチャ・メモリ・クラスタから

の色情報を、対応する描画プロセッサに出力する。描画プロセッサでは、予め入力されたピクセルの色情報とテクスチャ・メモリ・クラスタからの色情報を考慮して、対応するフレーム・バッファに格納されている該当ピクセルの値に反映させる。

【0037】なお、各テクスチャ・メモリ・クラスタ22～28には、予め全て同一の内容のテクスチャ・イメージ・データが重複して格納されている。従って、4つのテクスチャ・メモリ・クラスタ22～28による並列効果によって性能向上、特に処理速度の高速化を図ることができ、重複してデータを保持すれば高価なメモリの容量は増大する。

#### 【0038】

【発明が解決しようとする課題】本発明は、上記事実を考慮して、コンピュータ・グラフィクス・インタフェースにあるテクスチャ・マッピングの機能を効率良く実行させることができるコンピュータ・グラフィクス装置を得ることが目的である。

【0039】特に、必要とするメモリの容量を劇的に減少させても効率が許容範囲内の悪化で済むシステムを得ることが目的である。

#### 【0040】

【課題を解決するための手段】上記目的を達成する本発明は、複数のテクセルからなるテクスチャ・イメージ・データをインターリーブして重複がないように各々のメモリに格納する複数のテクスチャ・メモリ・クラスタであって、ある1つのピクセルに対するテクスチャ・イメージの座標から、その座標に対応するテクスチャ値を計算するのに必要なテクセルを求め、そのテクセルの色情報であるテクセル値を複数のテクスチャ・メモリ・クラスタから収集する収集手段と、収集したテクセル値から1のピクセルに対するテクスチャ値を計算する手段とをそれぞれ有する複数のテクスチャ・メモリ・クラスタと、複数のテクスチャ・メモリ・クラスタを接続するバスと、複数のテクスチャ・メモリ・クラスタのうちの1つにそれぞれ接続され、前述の1のピクセルに対するテクスチャ・イメージの座標を計算する複数のテクスチャ生成器と、複数のテクスチャ・メモリ・クラスタのうちの1つにそれぞれ接続され、前述の1のピクセルに対するテクスチャ値を用いて前述の1のピクセルに表示すべきデータを生成する複数の描画プロセッサとを有するテクスチャ・マッピングを行うコンピュータ・システムである。このようにすると、各テクスチャ・メモリ・クラスタ内のメモリに重複するテクセルを保持することなく、効率的にテクセル値を各テクスチャ・メモリ・クラスタ間で交換することができる。

【0041】また、上述の収集手段は、ある1つのピクセルに対するテクスチャ・イメージの座標から、その座標に対応するテクスチャ値を計算するのに必要なテクセルを求め、複数のテクスチャ・メモリ・クラスタに通知

する手段と、メモリを検索し、通知されたテクセルのテクセル値を得る検索手段と、テクセル値をテクセルの通知元のテクスチャ・メモリ・クラスタに送信する手段と、テクセル値を受信する手段とを含むようにすることも考えられる。

【0042】さらに、上述のメモリは複数のバンクに別れており、各バンクは、テクスチャ・メモリ・クラスタが格納するテクスチャ・イメージ・データをインターリーブして重複を生じないように格納するようにすることも考えられる。これにより、同一テクスチャ・メモリ・クラスタ内のメモリに同一テクセル値を重複して保持する、ということを完全に排除することができる。

【0043】また、先の複数のバンクは、入力FIFOバッファと、出力FIFOバッファとを含むことも考えられる。このようにすると、テクセルをそのアドレスでインターリーブして格納した場合に、同一バンクに対するアクセスの集中によるパフォーマンス低下を抑えることができる。

【0044】また、上述の検索手段は、通知されたテクセルのアドレスを計算する手段と、計算されたアドレスから該当するメモリのバンクにアドレスを分配する分配手段と、メモリのバンクからテクセル値を得る手段とを含むことも考えられる。このように分配することにより、効率よくメモリにアクセスすることができる。

【0045】さらに、先の必要なテクセルの通知元のテクスチャ・メモリ・クラスタのIDを記憶するFIFOバッファをテクスチャ・メモリ・クラスタの各々がさらに有するようすることも考えられる。このようにすると、必要なテクセルのテクセル値を必要なテクスチャ・メモリ・クラスタに送信することができる。

【0046】また、先の計算されたアドレスを分配した前記メモリのバンクのIDを記憶するFIFOバッファをテクスチャ・メモリ・クラスタの各々がさらに有するようすることも考えられる。このようにすると、検索結果を適切なメモリ・バンクから出力させることができる。

【0047】さらに、上述の収集手段が、ある1つのピクセルに対するテクスチャ・イメージの座標を複数のテクスチャ・メモリ・クラスタに通知する手段と、座標から、その座標に対応するテクスチャ値を計算するのに必要なテクセルであって、自己のメモリが保持するテクセルを計算する手段と、テクセルのテクセル値をメモリから取り出す手段と、テクセル値をテクセルの通知元のテクスチャ・メモリ・クラスタに送信する手段と、テクセル値を受信する手段とを含むようにすることも考えられる。

【0048】本発明の他の態様としては、複数のテクセルからなるテクスチャ・イメージと、そのテクスチャ・イメージを1以上の段階で縮小したデータであり且つ1又は複数のテクセルからなる縮小テクスチャ・イメージ

とを含むテクスチャ・イメージ・データをインターリーブして重複を生じないよう各々のメモリに格納する複数のテクスチャ・メモリ・クラスタであって、ある1つのピクセルに対するテクスチャ・イメージの座標及びピクセルとテクスチャ・イメージのテクセルの縮小率から前述の1のピクセルに対するテクスチャ値を計算するのに必要なテクセルを求め、そのテクセルの色情報であるテクセルの値を複数のテクスチャ・メモリ・クラスタが出力するように要求する手段と、要求されたテクセルの値を検索し、要求したテクスチャ・メモリ・クラスタに出力する手段と、テクセルの値を収集する手段と、収集されたテクセルの値からテクスチャ値を計算する手段とをそれぞれ有する先の複数のテクスチャ・メモリ・クラスタと、複数のテクスチャ・メモリ・クラスタを接続するバスと、複数のテクスチャ・メモリ・クラスタのうちの1つにそれぞれ接続され、前述の1のピクセルに対するテクスチャ・イメージの座標及び縮小率を計算する複数のテクスチャ生成器と、複数のテクスチャ・メモリ・クラスタのうちの1つにそれぞれ接続され、前述の1のピクセルに対するテクスチャ値を用いて前述の1のピクセルに表示すべきデータを生成する複数の描画プロセッサとを有するテクスチャ・マッピングを行うコンピュータ・システムがある。これによれば、ミップマップを用いるようなテクスチャ・マッピングにおいてテクスチャ・メモリ・クラスタのメモリを有効利用することができる。

#### 【0049】

【作用】第1に述べた本発明の構成では、各テクスチャ生成器が1のピクセルに対するテクスチャ・イメージの座標を生成して（ラスタライズ）、それぞれに接続されたテクスチャ・メモリ・クラスタに送る。そして、各テクスチャ・メモリ・クラスタでは、収集手段がそのテクスチャ・イメージの座標から必要なテクスチャ・イメージ・データのテクセルを計算する。そして、そのテクセルを有する、自己を含むテクスチャ・メモリ・クラスタからそのテクセルのテクセル値を収集する。必要なテクセル値を収集できれば、先の1のピクセルに対するテクスチャ値を計算する。計算されたテクスチャ値は、計算を行ったテクスチャ・メモリ・クラスタに接続された描画プロセッサで、その1のピクセルに表示すべきデータを計算する時に用いられる。そして、その表示すべきデータはフレーム・バッファに入れられる。

【0050】上述の収集手段が通知手段と検索手段と送信手段と受信手段に別れる場合には、通知手段が、必要なテクセルを自己を含む複数のテクスチャ・メモリ・クラスタに通知する。そして、検索手段がそのテクセルを有しているかメモリを検索し、有していればそのテクセルのテクセル値を出力する。出力があれば送信手段により要求元に送信し、要求元のテクスチャ・メモリ・クラスタの受信手段がそのテクセル値を受信する。

【0051】上述のようにメモリを複数のバンクに分けており、各バンクでもテクスチャ・イメージのインターリーブが行われている場合には、その各バンクに目的とするテクセルが格納されているかを記録しておき、そのバンクからデータを引き出す必要がある。

【0052】上述のように各メモリ・バンクに入力FIFOと出力FIFOを有していれば、必要なテクセル値が同一のメモリ・バンクに格納されている場合でも、一旦FIFOに格納しておくことにより、同一メモリ・バンクへのアクセスがあっても全体のパフォーマンスの低下を抑えることができる。

【0053】また、上述の検索手段が、アドレス計算手段と分配手段とテクセル値を得る手段とを含む場合には、通知されたテクセルのメモリ中のアドレスをそのアドレス計算手段が計算し、分配手段がそのアドレスに応じて該当メモリ・バンクにそのアドレスを分配する。そして、そのメモリ・バンクから目的のテクセル値を得ることができる。

【0054】さらに、必要なテクセルの通信元のテクスチャ・メモリ・クラスタのIDを記憶するFIFOバッファを有していれば、そのテクセルの要求を受信した際に要求したテクスチャ・メモリ・クラスタをこのFIFOに入力し、目的のテクセル値を格納したメモリ・バンクからの出力をそのFIFOの記憶内容の順（要求の受信順）に送信するようにできる。

【0055】計算されたアドレスを分配したメモリ・バンクのIDを記憶するFIFOバッファを有している場合には、アドレスの分配を行う際に分配したメモリ・バンクのIDをこのFIFOに記憶しておき、先のテクスチャ・メモリ・クラスタのFIFOと併せてどのバンク出力をどのテクスチャ・メモリ・クラスタに出力するかを対応付けする。

【0056】収集手段の他の態様の場合には、テクスチャ・イメージの座標をそのまま、自己を含む複数のテクスチャ・メモリ・クラスタに送信し、受信したテクスチャ・メモリ・クラスタで、計算手段が必要なテクセルであって保持しているテクセルを計算する。このテクセルのテクセル値を取り出し、要求元（通信元）のテクスチャ・メモリ・クラスタに送信する。そのテクスチャ・メモリ・クラスタは、送られてきたテクセル値を受信し、フィルタにてテクスチャ値を計算することになる。

【0057】さらに他の本発明の態様の場合には、テクスチャ生成器が、1のピクセルに対するテクスチャ・イメージの座標とそのピクセルとテクセルの縮小率を計算し、それを接続されているテクスチャ・メモリ・クラスタに送信する。そして、要求されたテクセルの値を検索し、要求したテクスチャ・メモリ・クラスタに送信する。そして必要なテクセル値を収集してテクスチャ値を計算する。計算されたテクスチャ値は、そのテクスチャ・メモリ・クラスタに接続された描画プロセッサによ



て、上述のピクセルの表示すべきデータの計算に用いられる。その表示すべきデータはフレーム・バッファに入力され、表示装置に表示される。

#### 【0058】

【実施例】図1に本願発明の用いられるシステムの全体を示す。ワークステーション1は、メインCPU、メイン・メモリ、キーボード、プリンタ等の入出力装置、及びFDDやHDDといった記憶装置を含む。このワークステーション1のバスに接続されているのが、グラフィック・サブシステム3であり、このサブシステム3の出力が表示装置11にて表示される。グラフィック・サブシステム3は、ジオメトリ計算部5とラスタ計算部7とテクスチャ計算部9により構成される。

【0059】この図1の動作を説明する。ワークステーション1のCPUは、表示装置11に何等かの表示を行いたい場合には、グラフィック・サブシステム3に描画を命令する。例えば、CPUがある位置（全体座標）における、ある物体に、あるテクスチャ・イメージを張り付けて描くことを命令した場合、ジオメトリ計算部5はポリゴン分割をし、ポリゴンの頂点におけるスクリーン上の座標を求め、その座標におけるテクスチャ・イメージの座標を計算する。もし、色の指定がさらにある場合にはその色情報も計算する。この結果を用いてラスタ計算部7が、そのポリゴン内部の各ピクセルごとにスクリーン上の座標及びテクスチャ・イメージの座標（あれば色情報も）を計算する。このテクスチャ・イメージの座標とテクセルの拡大縮小率LOD（縮小率ともいう。）からテクスチャ計算部9が、そのピクセルに対応するテクスチャの値を生成する。そして、そのテクスチャの値を用いて表示すべきデータを生成し、フレーム・バッファに格納する。格納されたデータにより表示装置11のスクリーン上に上述の物体を描き出すようになっている。

【0060】このグラフィック・サブシステムの詳細を図2に示す。ジオメトリ計算部5は、バス13により複数のテクスチャ生成器／フラグメント生成器15、17、19、21、23、25、27、29（ここでは8つ）に接続されている。そして、各テクスチャ生成器／フラグメント生成器は、1のテクスチャ・メモリ・クラスタに接続されている。このテクスチャ・メモリ・クラスタ33、35、37、39、41、45、47は、それぞれアドレス分配バス31とデータ交換バス95で接続されている。また、各テクスチャ・メモリ・クラスタは描画プロセッサ51、53、55、57、59、61、65にそれぞれ接続されている。この描画プロセッサは、フラグメント生成器も直接接続されている。そして各描画プロセッサは、フレーム・バッファ67、69、71、73、75、77、79、81にそれぞれ接続されている。

【0061】テクスチャ・メモリ・クラスタの概略はテ

クスチャ・メモリ・クラスタ33にのみ示してある。他のテクスチャ・メモリ・クラスタに関しては、その内容は同一であり説明を省略する。テクスチャ生成器15からの出力はアドレス生成器83に接続されており、アドレス生成器83の出力はアドレス分配バスに接続されている。また、他の出力はアドレス選択器85に接続されている。アドレス選択器85の出力は、アドレス分配器87に接続され、このアドレス分配器87は複数のメモリ・バンク89（ここでは8つ）に接続される。メモリ・バンク89の出力はデータ収集器91に接続されている。このデータ収集器91の出力はデータ交換バス95にも接続されており、他のテクスチャ・メモリ・クラスタにデータを出力するようになっている。このデータ収集器91の出力は、フィルタ93にも接続されている。フィルタ93では、このデータ収集器91からの出力とともに、データ交換バス95に接続されており、他のテクスチャ・メモリ・クラスタからの出力を受信するようになっている。そして、フィルタ93は生成したテクスチャ値を担当の描画プロセッサ51に出力するようになっている。

【0062】以下、図2に示した装置の動作を説明する。ジオメトリ計算部5は、上述したように各ポリゴンの頂点に係るピクセルの座標とテクスチャ・イメージの座標、その点での色情報を計算し出力する。この出力はバス13でブロードキャストされ、各テクスチャ生成器／フラグメント生成器で受信される。各テクスチャ生成器はポリゴン内部のピクセルを補間計算し、各ピクセルごとの、スクリーン上の座標、テクスチャ・イメージの座標を計算する。また、フラグメント生成器は、同様にポリゴン内部のピクセルを補間計算し、各ピクセルごとの、スクリーン上の座標、色情報を計算する。この補間計算の際には、ピクセルとテクセルの縮小率LODも同時に計算される。但し、各テクスチャ生成器及びフラグメント生成器は担当するピクセルを予め割り当てられており、担当するピクセル以外のデータは生成されない。なお、ラスタ計算器7は図2では、テクスチャ生成器／フラグメント生成器に対応する。

【0063】ここで、各テクスチャ・メモリ・クラスタがどのようにテクスチャ・イメージを保持しているかを簡単に説明する。なお、一般のアプリケーションでは従来技術で述べたリニア・ミップマップ・リニアが用いられており、この方式でテクスチャ・マッピングを行う際にパフォーマンスを最大にすることを目的とした例を述べることを予め断っておく。このリニア・ミップマップ・リニアでは、同時に8つのテクセル値にアクセスできなければならない。よって、必要なテクセル値が8つのテクスチャ・メモリ・クラスタに分散されて記憶されているようにする。すなわち図2の例では、例えばテクスチャ・メモリ・クラスタ33（クラスタ0）は縮小率LODが偶数で、iが偶数、jが偶数のテクセル、テクス

チャ・メモリ・クラスタ35（クラスタ1）は縮小率L  
ODが偶数で、iが奇数、jが偶数のテクセル、テクス  
チャ・メモリ・クラスタ37（クラスタ2）は縮小率L  
ODが偶数で、iが偶数、jが奇数のテクセル、テクス  
チャ・メモリ・クラスタ39（クラスタ3）は縮小率L  
ODが偶数で、iが奇数、jが奇数のテクセル、テクス  
チャ・メモリ・クラスタ41（クラスタ4）は縮小率L  
ODが奇数で、iが偶数、jが偶数のテクセル、テクス  
チャ・メモリ・クラスタ43（クラスタ5）は縮小率L  
ODが奇数で、iが奇数、jが偶数のテクセル、テクス  
チャ・メモリ・クラスタ45（クラスタ6）は縮小率L  
ODが奇数で、iが偶数、jが奇数のテクセル、テクス  
チャ・メモリ・クラスタ47（クラスタ7）は縮小率L  
ODが奇数で、iが奇数、jが奇数のテクセル、とす  
る。

【0064】元のテクスチャ・イメージが8×8の場合  
の例を、図3に示す。図3（A）は、LODが0のテク  
スチャ・イメージを示している。この各枠内の数字は、  
そのテクセル値が格納されるテクスチャ・メモリ・クラ  
スタの番号（上記のカッコ内の数字）を表す。このよう  
にLODが0、すなわち偶数の場合には、クラスタ0か  
らクラスタ3までに格納される。またLODが1（図3  
（B））の場合には、クラスタ4からクラスタ7までに  
格納される。図3（C）の場合も同様である。

【0065】このようにテクスチャ・イメージが各テク  
スチャ・メモリ・クラスタに格納されていることを前提  
に以下動作の説明を続行する。テクスチャ生成器からテ  
クスチャ・イメージの座標及び縮小率LODを受信した  
テクスチャ・メモリ・クラスタは、アドレス生成器が必要  
なテクセルを見出す。すなわち、計算されたLODの  
整数部分の値とそれに+1した値（必ず偶数と奇数に  
なる）と、そのレベル毎のテクセルの座標（i, j）を  
計算する。そして、それをアドレス分配バス31に出力  
するとともに、接続されたアドレス選択器にも出力す  
る。アドレス分配バス31の出力も、それを出力したテ  
クスチャ・メモリ・クラスタ以外のクラスタのアドレス  
選択器に入力される。例えば、アドレス生成器が計算し  
たテクセルの座標を左下として、1つのLODのミップ  
マップにつき4つのテクセル値を参照するので、各アド  
レス選択器は受信した座標のテクセルではなく、その4  
つの内の自己が保持しているテクセルが要求されてい  
ると認識し、そのテクセルの座標を計算する。他のLOD  
の4つのテクセルについても同様である。但し、ここ  
ではアドレス選択器で計算していた処理は、要求元のアド  
レス生成器が1つのLODごとに4つのテクセルの座標  
を生成し、保持しているテクスチャ・メモリ・クラス  
タに送信しても同様である。

【0066】このようにして、各アドレス選択器には、  
直接接続されたアドレス生成器からの要求も含めて、8  
つのテクセルについての要求が入力される。そして、こ

のアドレス選択器は8つの要求を順にアドレス分配器に  
出力する。アドレス分配器は、テクセルの座標に対応す  
る、メモリの物理アドレスを計算し、その物理アドレス  
に対応するメモリ・バンクにその物理アドレスを出力す  
る。そして目的とするテクセル値がデータ収集器に出力  
され、要求元のテクスチャ・メモリ・クラスタにデータ  
交換バスにて送られる。もし、そのテクセル値が自己の  
テクスチャ・メモリ・クラスタの要求であった場合に  
は、フィルタに直接送られる。フィルタは、上述のよう  
にデータ交換バスにて送られてきた7つのテクセル値  
と、自己のデータ収集器から送られてきた1つのテクセ  
ル値で所定の計算を行い、テクスチャ値を生成する。生  
成されたテクスチャ値は、接続された描画プロセッサに  
出力される。描画プロセッサは、そのテクスチャ値が対  
応するスクリーン上のピクセルに対応する、フラグメン  
ト生成器により生成された色情報と、フレーム・バッフ  
ァの値を参照して、そのピクセルに関する表示データを  
生成し、所定のフレーム・バッファに格納する。フレー  
ム・バッファに入力されたデータは、コントローラによ  
り読み出され、表示装置に表示される。

【0067】以上、図2の動作を述べたが、メモリ・バ  
ンクについてさらに述べる。上で述べたように、アドレ  
ス分配器は、物理アドレスに対応するメモリ・バンクに  
その物理アドレスを出力する。すなわち、各テクスチャ  
・メモリ・クラスタ内のメモリ・バンクでも、物理アド  
レスでインターリーブして記憶している。ここではメモ  
リ・バンクは8つであるから、物理アドレスの下位3桁  
で、インターリーブするとよい。このようにすれば、1  
つのテクセル値も重複することなく、テクスチャ・イメ  
ージをテクスチャ・メモリ・クラスタに格納することが  
でき、メモリ容量の削減に有効である。例えば、第1の  
従来例では、通信して各プロセッサで輝度計算を行うた  
め、各プロセッサ内のメモリに格納されるテクセルに重  
複が必ず生じていた。また、第2の従来例でも、1Mb  
yteのテクスチャ・イメージを格納する際には8Mb  
yteのメモリが必要になる。これに対し、本発明は1  
Mbyteのテクスチャ・イメージを格納するのに1M  
byteしか必要なく、大容量化しつつあるテクスチャ  
・イメージを大量に格納することができ、有効である。

【0068】図2に示したテクスチャ・メモリ・クラス  
タの詳細を図4に示す。テクスチャ生成器の出力は、偶  
LODアドレス生成器83aと奇LODアドレス生成器  
83bに接続されている。また、この2つのアドレス生  
成器の出力は、アドレス分配バス31に接続される。さ  
らに、そのテクスチャ・メモリ・クラスタの格納してい  
るテクセルに依存して、2つのアドレス生成器のどちら  
かの出力が、アドレス選択器85に接続される。このア  
ドレス選択器85には、バッファ109を介して他のテ  
クスチャ・メモリ・クラスタからの出力が入力されるよ  
うになっている。そして、アドレス選択器85はアドレ

ス分配器87に接続されており、このアドレス分配器87は複数のメモリ・バンク105（ここでは、8つ）に接続されている。各メモリ・バンクには、入力FIFOバッファ105aとDRAM105cと出力FIFOバッファ105bがある。またアドレス選択器87は、タグFIFOバッファ115とあて先FIFOバッファ103に接続されている。このメモリ・バンク及びタグFIFO115及びあて先FIFO103の出力はデータ収集器91に接続され、このデータ収集器91の出力は、一方はバッファ113を介して他のテクスチャ・メモリ・クラスタに出力され、他方はフィルタ93に出力されるようになっている。フィルタ93には他のテクスチャ・メモリ・クラスタからの出力がバッファ111を介して入力されるようになっている。フィルタ93の出力は、上述のように描画プロセッサに接続されている。

【0069】以下、図4の動作を説明する。図2のアドレス生成器は、偶LODアドレス生成器83aと奇LODアドレス生成器83bで構成されている。この偶LODアドレス生成器は、偶数のLODに係るテクスチャ・イメージを格納するテクスチャ・メモリ・クラスタに接続されている。同様に、奇LODアドレス生成器は、奇数のLODに係るテクスチャ・イメージを格納するテクスチャ・メモリ・クラスタに接続されている。当然、自己のテクスチャ・メモリ・クラスタもテクセルを格納しているから、自己の格納しているテクスチャ・イメージのLODによりマルチプレクサ101を操作して、アドレス選択器85に出力する。そして、テクスチャ値を計算するのに必要な定数を定数FIFO107に格納する。必要な定数は従来技術の説明でa, b, dと言っていたものである。

【0070】他のテクスチャ・メモリ・クラスタでも同様にして、例えば必要なテクセルの左下のテクセルの座標を出力するので、その出力がバッファ109に入力される。アドレス選択器は、バッファ109及びマルチプレクサ101の出力から1つずつ選択してきて、先ほど述べたように、左下のテクセルの座標に対応し且つ必要とする4つのテクセルの内の自己の格納しているテクセルを認識する。そしてそのテクセルの座標を計算する。但し、左下のテクセルの座標を得た段階で、必要なテクセルの格納された物理アドレスを出力するような構成にすれば、より効率的である。図2の説明では、話の流れ上前者のような動作であるとしたが、対応関係は明瞭であるから後者のようにすることは何等の困難もない。アドレス選択器85で生成された物理アドレスは、アドレス分配器がそのアドレスの下位3桁で、分配するメモリ・バンクを認識し、そのメモリ・バンクのID（又はタグ）をタグFIFO115に入力する。また、その物理アドレスに格納されたテクセル値を要求したテクスチャ・メモリ・アドレスのIDをあて先FIFO103に入力する。これにより、どのメモリ・バンクからの出力

が、どのテクスチャ・メモリ・クラスタに出力されるかが順番を誤ることなく認識できる。物理アドレスは対応するメモリ・バンクの入力FIFOバッファ105aに入力される。このFIFOは、アドレスによりインターリーブしているために、入力される物理アドレスがある1つのメモリ・バンクに集中する場合がある。このような場合であっても、順番を誤ることなく、且つあふれることなく、出力を得るために設けたものである。出力FIFOバッファ105bも同様の理由で設けられるものである。

【0071】但し、入力FIFOを設けたとしてもあふれる場合も考えられるので、あふれたかどうかはアドレス分配器は検査する。もしあふれた場合には、そのメモリ・バンクの処理が進み、入力FIFOバッファ105aに空きが生ずるまで、入力FIFOバッファ105aへの入力を停止する。そして、テクセル値は出力FIFOバッファ105bに入力される。ここでデータ収集器91は、タグFIFO115とあて先FIFO103を1つずつそれぞれ読み出す。そして、タグFIFO115の示すメモリ・バンクの出力FIFO105bからテクセル値を読み出し、対応するバッファ113に出力する。または、自己の要求に基づく場合にはフィルタ93に出力する。そして、バッファ113から要求元のテクスチャ・メモリ・クラスタに出力される。

【0072】このようにしてデータ交換バスを介して得たテクセル値は、まず、バッファ111に格納され、そしてフィルタ93に入力される。これにより8つの必要なテクセル値が集まったので、定数FIFO107から必要な定数を読み出し、従来技術の説明で示した計算をフィルタ93が行う。この計算によりテクスチャ値が生成され、描画プロセッサに与えられる。

【0073】以上述べたように、これによりメモリの有効利用を行うと共に、この例ではラスタ計算部以下を並列に行うことにより、テクスチャ・メモリ・クラスタ間の通信を行うにしても、パフォーマンスを許容範囲内の悪化にとどめることができた。しかし、これは一例であって、例えば、テクスチャ・メモリ・クラスタの数はその同時アクセスが必要なテクセルの数に合わせることで、又はパフォーマンスを落とすのであれば複数回テクスチャ・メモリ・アクセスするようにして、テクスチャ・メモリ・クラスタの数を減らす等の変更を行うことができる。さらに、アドレス生成器をテクスチャ生成器に接続している例を示したが、縮小率LODとテクスチャ・イメージの座標をそのまま複数のテクスチャ・メモリ・クラスタにブロードキャストし、アドレス選択器で計算してもよい。また、メモリ・バンクの数も、アクセスが集中しないような数に変更することは可能である。さらに、入力FIFOバッファ及び出力FIFOバッファの段数も、同様にメモリ・バンクの容量等に依存するので、変更可能である。

【0074】さらに、あて先FIFO及びタグFIFOの段数もメモリ・アクセスの速度等に依存するので、変更可能である。また、8つのテクセルを同時に必要とする例を示したが、この数が増加した場合、減少した場合にも、そのインターリーブの仕方及びテクスチャ・メモリ・クラスタの数にて対応可能である。

【0075】

【発明の効果】以上説明したように、コンピュータ・グラフィクス・インタフェースにあるテクスチャ・マッピングの機能を効率良く実行させることができるコンピュータ・グラフィクス装置を得ることができた。

【0076】また、必要とするメモリの容量を劇的に減少させても効率が許容範囲内の悪化で済むシステムを得ることもできた。

【0077】具体的には、メモリ・バンクの数を8、物理アドレスはランダムである、DRAMのアクセス時間は60nsである、各FIFOは7.5nsで動作可能、物理アドレスは7.5nsごとに入力され、アドレス分配器は7.5nsごとに分配することができる。データ収集器は7.5nsで出力FIFOからデータを回収できる、との条件でシミュレーションを行った結果、DRAMの利用率は85パーセントとなり、アクセスレートとしては約113MHzであった。

【図面の簡単な説明】

【図1】本発明が適用可能される装置の全体図である。

【図2】本発明が適用可能な一実施例のコンピュータ・グラフィクス装置の要部構成を示すブロック図である。

【図3】テクスチャ・メモリ・クラスタに記憶するテクセルの格納方法を示す概念図である。

【図4】テクスチャ・メモリ・クラスタ内の詳細なブロック図である。

【図5】コンピュータ・グラフィクスにおいて用いる座標系を示す線図である。

\*【図6】ミップマップを説明するための説明図であり、(A)はレベル0のミップマップであるテクスチャイメージを示す線図、(B)はサイズが半分のレベル1のミップマップであるテクスチャ示す線図である。

【図7】従来のコンピュータグラフィクス装置の要部構成を示すブロック図である。

【図8】従来例のデータ・フォーマットを示すブロック図である。

10 【図9】従来例のデータ・フォーマットを示すブロック図である。

【図10】従来例のデータ・フォーマットを示すブロック図である。

【符号の説明】

1 ワークステーション 3 グラフィック・サブシステム

5 ジオメトリ計算部 7 ラスタ計算部

9 テクスチャ計算部 11 表示装置

13 バス

15から29 テクスチャ生成器/フラグメント生成器

31 アドレス分配バス

33から47 テクスチャ・メモリ・クラスタ

51から65 描画プロセッサ

67から81 フレーム・バッファ

83 アドレス生成器 85 アドレス選択器

87 アドレス分配器 89 メモリ・バンク

91 データ収集器 93 フィルタ

95 データ交換バス 101 マルチプレクサ

103 あて先FIFO 105 メモリ・バンク

30 107 定数FIFO 109 バッファ

111 バッファ 113 バッファ

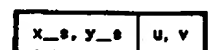
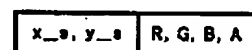
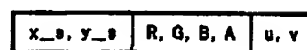
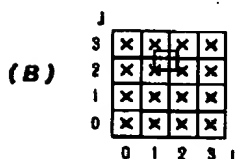
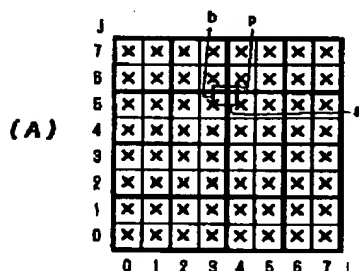
\* 115 タグFIFO

【図6】

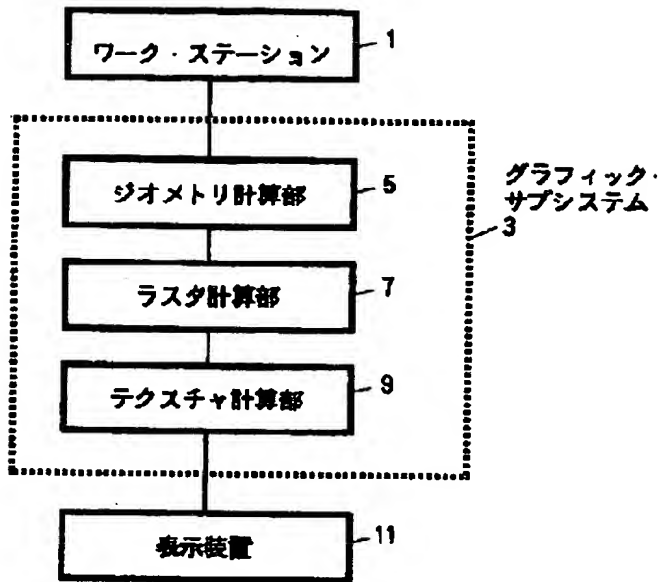
【図8】

【図9】

【図10】



【図1】



【図3】

Figure 3 shows three data structures: LOD0, LOD1, and LOD2.

**LOD0**

J	7	2	3	2	3	2	3	2	3
6	0	1	0	1	0	1	0	1	
5	2	3	2	3	2	3	2	3	
4	0	1	0	1	0	1	0	1	
3	2	3	2	3	2	3	2	3	
2	0	1	0	1	0	1	0	1	
1	2	3	2	3	2	3	2	3	
0	0	1	0	1	0	1	0	1	
	0	1	2	3	4	5	6	7	

**(A)**

**LOD1**

J	3	6	7	6	7
2	4	5	4	5	
1	6	7	6	7	
0	4	5	4	5	
	0	1	2	3	

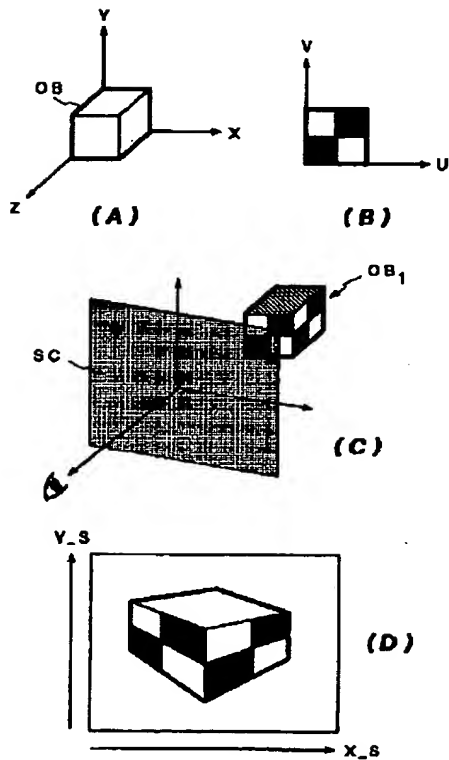
**(B)**

**LOD2**

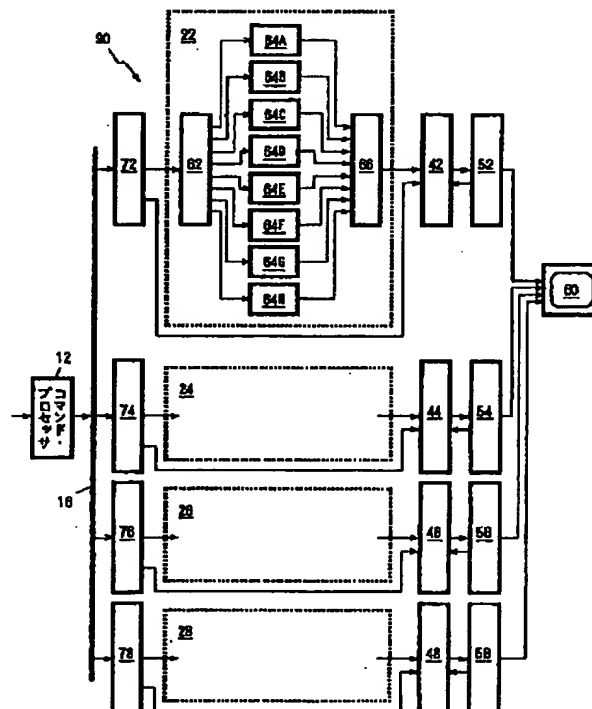
J	1	2	3
0	0	1	
	0	1	

**(C)**

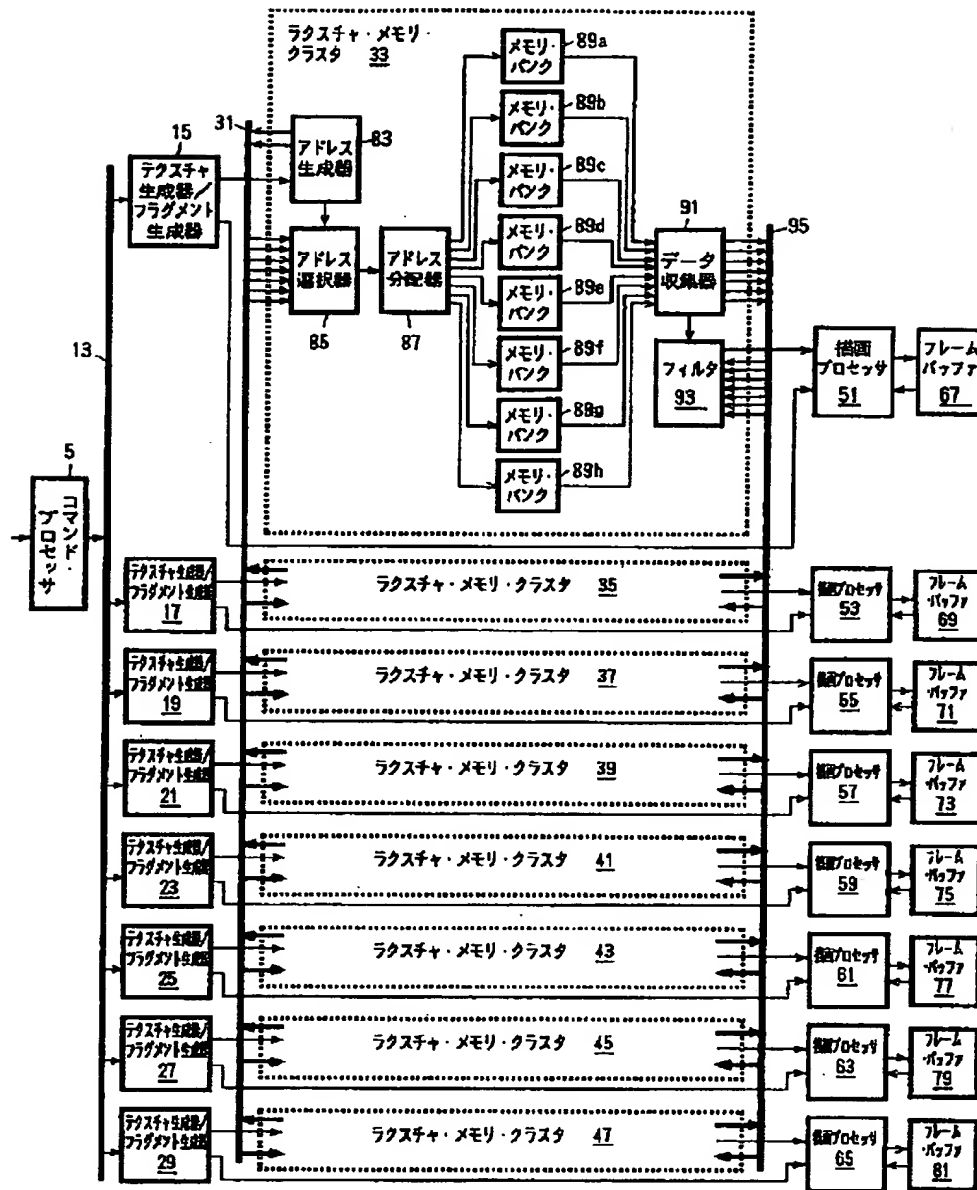
【図5】



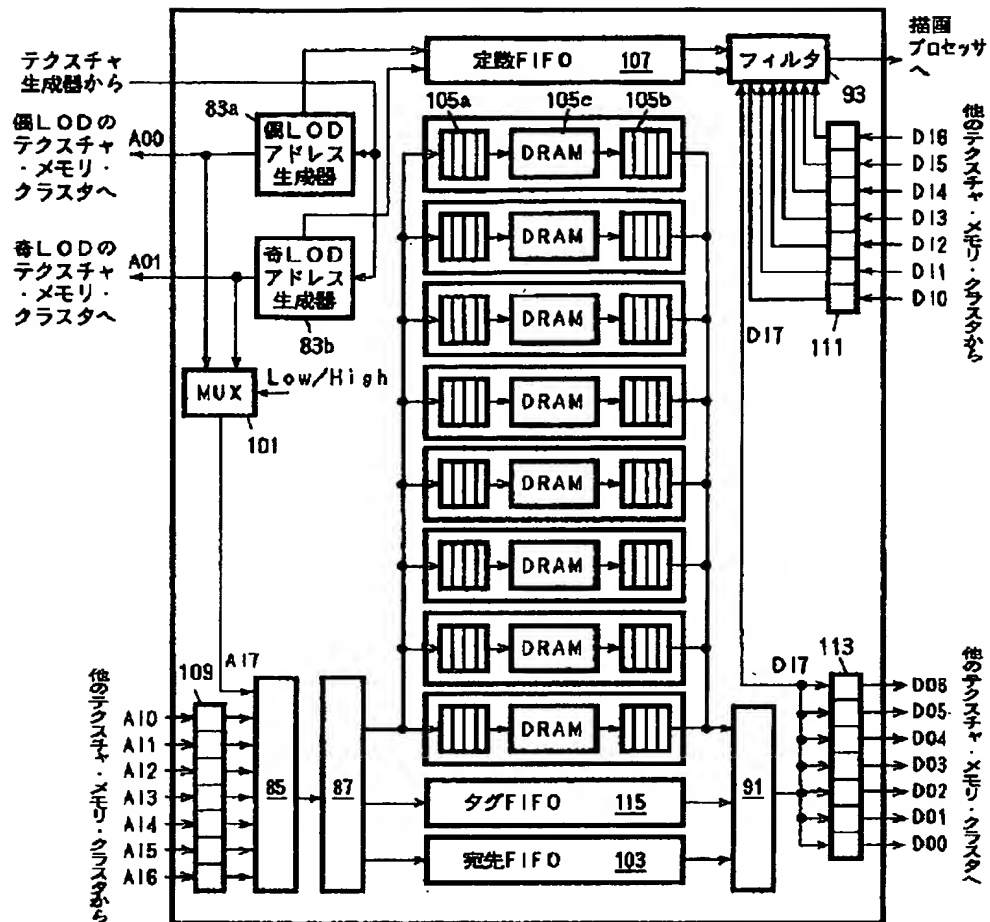
【図7】



【図 2】



【図 4】



フロントページの続き

(72)発明者 田中 伸宜

神奈川県大和市下鶴間1623番地14 日本ア  
イ・ビー・エム株式会社 大和事業所内

**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** \_\_\_\_\_

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**